Science of Computer Programming 8 (1987) 105-112
North-Holland

SMALL PROGRAMMING EXERCISES 15

M. REM

Department of Mathematics and Computing Science, Eindhoven University of Technology, 5600 MB Eindhoven, Netherlands

The testing of digital circuits, such as silicon chips, is a difficult subject. Even under rather restrictive conditions the problem of finding suitable test patterns is NP-complete. For Exercise 38 I have carefully chosen a testing problem that is not NP-complete. As a matter of fact, it has a linear solution. The problem is to determine the number of input patterns for which a tree of (possibly malfunctioning) NAND-gates produces an erroneous result.

This exercise is preceded by one in which all gates function correctly. The circuit is now an arbitrary combinational, i.e. acyclic, network of XOR-gates. We are requested to write a program that determines all inputs on which the circuit's output depends.

Exercise 37: Acyclic XOR-composition

A *source* of a directed graph is a vertex without incoming arcs, a *sink* is a vertex without outgoing arcs. Given is an acyclic directed graph G in which multiple arcs are allowed. Graph G has one sink and all non-sources have two incoming arcs. It represents a network of XOR-gates. If we assign to each source a value 0 or 1 every vertex produces a value: each source produces the value assigned to it, and each non-source produces value $(x+y)$**mod**2, where x and y are the values produced by its predecessors. The value produced by the sink is considered the value "computed by G".

Function

$$xor\colon (x, y) \rightarrow (x+y) \textbf{ mod } 2$$

is associative. It may, therefore, be generalized to m, $m \geq 0$, arguments. The 1-argument *xor* is the identity function; the 0-argument *xor* yields value 0. For every vertex there is a set of sources such that the vertex produces the (generalized) *xor* of the values assigned to those sources. We wish to determine this set of sources for the sink. The set may be empty: for example, the sink of a graph that consists of just two vertices with two arcs between them produces value 0 for all assignments to the source, i.e., it produces the *xor* of the empty set.

Graph G has N, $N \geq 1$, vertices, which are numbered from 0 upward. The sources are numbered 0 through $M-1$. Vertex $N-1$ is the (only) sink of G. The graph is

given by two integer arrays, $p0(i: M \leq i < N)$ and $p1(i: M \leq i < N)$: vertex i, $M \leq i < N$, has $p0(i)$ and $p1(i)$ as its (not necessarily distinct) predecessors. We have to determine S in

$$
\begin{array}{l}
|[M, N: int; \{1 \leq M \leq N\} \\
\quad p0, p1(i: M \leq i < N): \textbf{array of } int; \\
\quad \{p0 \text{ and } p1 \text{ represent graph } G \text{ with sink } N-1\} \\
\quad |[b(i: 0 \leq i < M): \textbf{array of } bool; \\
\quad\quad S \\
\quad\quad \{\text{the sink of } G \text{ produces the } xor \text{ of the values assigned} \\
\quad\quad\quad \text{to the set } \{i \mid 0 \leq i < M \wedge b(i)\} \text{ of sources of } G\} \\
\quad]| \\
]|
\end{array}
$$

Exercise 38: Test patterns for tree compositions

Given is a rooted tree with its arcs directed towards the root, in which each non-leaf has two predecessors. It represents a network of NAND-gates. As in Exercise 37, each leaf produces the value (0 or 1) assigned to it. Each NAND-gate produces value $1 - x \cdot y$, where x and y are the values produced by its two predecessors. We are, of course, interested in the value produced by the root.

In this exercise vertices may malfunction: each vertex (including the leaves and the root) may be stuck at 0 or stuck at 1. If a vertex is stuck at v it always produces value v. Due to this malfunctioning, the root may, for certain assignments to the leaves, produce the wrong value. Such an assignment is called a *test pattern*. For example, a tree of three vertices of which the root is stuck at 0 has three test patterns: (0, 0), (0, 1), and (1, 0). If its root is stuck at 1 it has only (1, 1) as a test pattern. We wish to determine the number of test patterns.

The tree has N, $N \geq 1$, vertices of which M, numbered 0 through $M-1$, are leaves ($N = 2 \cdot M - 1$). Vertex $N-1$ is the root. Array $s(i: 0 \leq i < N-1)$ represents the tree: vertex i has vertex $s(i)$ as its successor. Array $f(i: 0 \leq i < N)$ records for each vertex i whether it is malfunctioning:

$$
\begin{array}{l}
-1 \leq f(i) \leq 1 \\
\wedge f(i) = -1 \Rightarrow (\text{vertex } i \text{ functions correctly}) \\
\wedge f(i) = 0 \Rightarrow (\text{vertex } i \text{ stuck at 0}) \\
\wedge f(i) = 1 \Rightarrow (\text{vertex } i \text{ stuck at 1}).
\end{array}
$$

We can now formulate the functional specification:

$$
\begin{array}{l}
|[M, N: int; \{N \geq 1 \wedge N = 2 \cdot M - 1\} \\
\quad s(i: 0 \leq i < N-1): \textbf{array of } int; \\
\quad \{(\mathbf{A}i: 0 \leq i < N-1: (\mathbf{E}j: j \geq 1: s^j(i) = N-1)) \\
\quad\quad \wedge (\mathbf{A}j: M \leq j < N: (\mathbf{N}i: 0 \leq i < N-1: s(i) = j) = 2)\} \\
\quad f(i: 0 \leq i < N): \textbf{array of } int; \\
\quad \{(\mathbf{A}i: 0 \leq i < N: -1 \leq f(i) \leq 1)\}
\end{array}
$$

```
|[k: int;
  S
  {k = (number of test patterns)}
 ]|
]|
```

Solution of Exercise 34 (diameter of a polygon)

We have to determine S in

```
|[N: int; {N ≥ 3}
  d(i: 0 ≤ i < N): array of int;
  {(Ai: 0 ≤ i < N: d(i) ≥ 1)}
  |[p, q: int;
    S
    {0 ≤ p < q < N
     ∧ abs(D(p, q) − D(q, p))
     = (MINi, j: 0 ≤ i < j < N: abs(D(i, j) − D(j, i)))}
  ]|
]|
```

where for $0 \leq i < N \wedge 0 \leq j < N$,

$$D(i,j) = \begin{cases} (\mathbf{S}h\colon i \leq h < j\colon d(h)) & \text{if } i \leq j, \\ (\mathbf{S}h\colon i \leq h < N\colon d(h)) + (\mathbf{S}h\colon 0 \leq h < j\colon d(h)) & \text{if } i > j. \end{cases}$$

Let

$$E = (\mathbf{S}h\colon 0 \leq h < N\colon d(h)).$$

Since

$$D(i,j) + D(j,i) = E$$

for $i \neq j$, we may replace the postcondition by

$$0 \leq p < q < N \wedge abs(2 \cdot D(p,q) - E) = (\mathbf{MIN}i, j\colon 0 \leq i < j < N\colon abs(2 \cdot D(i,j) - E))$$

We define for $0 \leq m \leq n \leq N$

$$C(m,n) = (\mathbf{MIN}i, j\colon m \leq i < j \wedge n \leq j < N\colon abs(2 \cdot D(i,j) - E)).$$

The postcondition then reads

$$0 \leq p < q < N \wedge abs(2 \cdot D(p,q) - E) = C(0,0).$$

Since

$$abs(2 \cdot D(p,p) - E) = E > C(0,0),$$

the postcondition is equivalent to

$$R\colon \quad 0 \leq p \leq q < N \wedge abs(2 \cdot D(p,q) - E) = C(0,0).$$

Inspired by R, we adopt $P0$ as the invariant:

$$P0: \quad 0 \leq p \leq q < N \wedge 0 \leq m \leq n \leq N \\ \wedge\ abs(2 \cdot D(p, q) - E) \textbf{ min } C(m, n) = C(0, 0).$$

Since $C(m, N) = \inf$, $P0 \wedge n = N$ implies R. We choose $n \neq N$ as the guard of the repetition. On account of $E > C(0, 0)$, invariant $P0$ is established by $p, q, m, n := 0, 0, 0, 0$.

In the repetition we want to increase m or n. For increasing m we look at the relation between $C(m, n)$ and $C(m+1, n)$:

$$C(m, n) = C(m+1, n) \textbf{ min}(\textbf{MIN} j: n \leq j < N: abs(2 \cdot D(m, j) - E)).$$

For $m \leq j < N$ function $2 \cdot D(m, j) - E$ is increasing in j. Hence, if $2 \cdot D(m, n) - E \geq 0$ we have

$$C(m, n) = C(m+1, n) \textbf{ min } abs(2 \cdot D(m, n) - E).$$

Consequently, if $P0 \wedge 2 \cdot D(m, n) - E \geq 0$ then

$$abs(2 \cdot D(p, q) - E) \textbf{ min } abs(2 \cdot D(m, n) - E) \textbf{ min } C(m+1, n) = C(0, 0). \tag{1}$$

The relation between $C(m, n)$ and $C(m, n+1)$ is the following:

$$C(m, n) = C(m, n+1) \textbf{ min}(\textbf{MIN } i: m \leq i < n: abs(2 \cdot D(i, n) - E)).$$

For $i < n$ function $2 \cdot D(i, n) - E$ is decreasing in i. Hence, if $2 \cdot D(m, n) - E \leq 0$ we have

$$C(m, n) = C(m, n+1) \textbf{ min } abs(2 \cdot D(m, n) - E)$$

Consequently, if $P0 \wedge 2 \cdot D(m, n) - E \leq 0$ then

$$abs(2 \cdot D(p, q) - E) \textbf{ min } abs(2 \cdot D(m, n) - E) \textbf{ min } C(m, n+1) = C(0, 0) \tag{2}$$

Notice that (1) and (2) are almost identical. They give rise to a solution of the form

```
p, q, m, n := 0, 0, 0, 0
;do n ≠ N
→ if abs(2 · D(p, q) − E) ≥ abs(2 · D(m, n) − E) → p, q := m, n
  □ abs(2 · D(p, q) − E) ≤ abs(2 · D(m, n) − E) → skip
  fi
; if 2 · D(m, n) − E ≥ 0 → m := m + 1
  □ 2 · D(m, n) − E ≤ 0 → n := n + 1
  fi
od
```

In view of the guards above, we extend the invariant with

$$P1: \quad e = 2 \cdot D(m, n) - E \wedge f = abs(e) \\ \wedge\ g = abs(2 \cdot D(p, q) - E)$$

Our final solution is then

S: $|[\ m, n, e, f, g: int;$
$\quad p, q, m, n := 0, 0, 0, 0$
$\quad ; |[\ i: int;\ e, i := 0, 0;\ \mathbf{do}\ i \neq N \rightarrow e, i := e - d(i), i + 1\ \mathbf{od}\ ]|$
$\quad ; f, g := -e, -e$
$\quad ; \mathbf{do}\ n \neq N$
$\quad\quad \rightarrow \mathbf{if}\ g \geqslant f \rightarrow p, q, g := m, n, f \ \square\ g \leqslant f \rightarrow \mathbf{skip}\ \mathbf{fi}$
$\quad\quad ;\ \mathbf{if}\ e \geqslant 0 \rightarrow e, m := e - 2 * d(m), m + 1$
$\quad\quad\quad \square\ e \leqslant 0 \rightarrow e, n := e + 2 * d(n), n + 1$
$\quad\quad\quad \mathbf{fi}$
$\quad\quad ;\ \mathbf{if}\ e \leqslant 0 \rightarrow f := -e \ \square\ e \geqslant 0 \rightarrow f := e\ \mathbf{fi}$
$\quad\quad \mathbf{od}$
$\quad]|$

The program has an execution time that is linear in N.

Solution of Exercise 35 (inscribed rectangle)

With D as defined in Exercise 34 we have to solve S in

$|[\ N: int;\ \{N \geqslant 4\}$
$\quad d(i: 0 \leqslant i < N): \mathbf{array\ of}\ int;$
$\quad \{(\mathbf{A}i: 0 \leqslant i < N: d(i) \geqslant 1)\}$
$\quad |[\ b: bool;$
$\quad\quad S$
$\quad\quad \{b = (\mathbf{E}p, q, r, s: 0 \leqslant p < q < r < s < N$
$\quad\quad\quad\quad\quad\quad : D(p, q) = D(r, s) \wedge D(q, r) = D(s, p))\}$
$\quad]|$
$]|$

The quantified expression in the postcondition may be written as

$$D(p, q) + D(q, r) = D(r, s) + D(s, p)$$
$$\wedge\ D(q, r) + D(r, s) = D(s, p) + D(p, q)$$

or

$$D(p, r) = D(r, p) \wedge D(q, s) = D(s, q)$$

or

$$2 \cdot D(p, r) - E = 0 \wedge 2 \cdot D(q, s) - E = 0,$$

with E (the circumference of the circle) defined as in the solution of Exercise 34. We may, consequently, change the postcondition into

R: $\quad b = ((\mathbf{N}i, j: 0 \leqslant i < j < N: 2 \cdot D(i, j) - E = 0) \geqslant 2).$

In words: there exists an inscribed rectangle if there are at least two pairs of diametrical points.

We design a repetition for counting the number of diametrical points. As the reader may expect, this will very much resemble the solution of the preceding exercise. We introduce as our invariant:

$$P: \quad 0 \leq m \leq n \leq N \wedge a + C(m, n) = C(0, 0) \wedge e = 2 \cdot D(m, n) - E$$

where

$$C(m, n) = (\mathbf{N}i, j: m \leq i < j \wedge n \leq j < N: 2 \cdot D(i, j) - E = 0).$$

The postcondition is

$$b \equiv (C(0, 0) \geq 2).$$

Since $C(m, N) = 0$, we have

$$P \wedge n = N \Rightarrow a = C(0, 0).$$

Furthermore,

$$P \wedge a \geq 2 \Rightarrow C(0, 0) \geq 2.$$

Hence,

$$P \wedge (n = N \vee a \geq 2) \Rightarrow (C(0, 0) \geq 2 \equiv a \geq 2).$$

We take $n \neq N \wedge a < 2$ as the guard. Upon termination of the repetition postcondition R may then be established by $b := (a \geq 2)$.

Using

$$C(m, n) = \begin{cases} C(m+1, n) & \text{if } 2 \cdot D(m, n) - E > 0, \\ C(m+1, n) + 1 & \text{if } 2 \cdot D(m, n) - E = 0, \end{cases}$$

and

$$C(m, n) = \begin{cases} C(m, n+1) & \text{if } 2 \cdot D(m, n) - E < 0, \\ C(m, n+1) + 1 & \text{if } 2 \cdot D(m, n) - E = 0, \end{cases}$$

we obtain the following solution:

```
S:  |[m, n, a, e: int;
      m, n, a := 0, 0, 0
    ; |[i: int; e, i := 0, 0; do i ≠ N → e, i := e − d(i), i + 1 od]|
    ; do n ≠ N ∧ a < 2
          → if e > 0 → e, m := e − 2 * d(m), m + 1
            □ e = 0 → e, m, n, a := e − 2 * d(m) + 2 * d(n), m + 1, n + 1, a + 1
            □ e < 0 → e, n := e + 2 * d(n), n + 1
            fi
      od
    ; b := (a ≥ 2)
    ]|
```

We have again found a linear solution without introducing auxiliary arrays.

Solution of Exercise 36 (weaving)

The projection $t\restriction A$ of a sequence t of numbers on a set A of numbers is the sequence obtained from t by deleting all numbers not in A. We have to find an S such that

$$
\begin{array}{l}
|[M, N: int; \{M \geq 1 \wedge N \geq 1\} \\
\quad u(i: 0 \leq i < M): \textbf{array of } int; \\
\quad v(j: 0 \leq j < N): \textbf{array of } int; \\
\quad \{u\restriction\{0,1\} = u \wedge v\restriction\{1,2\} = v \wedge u\restriction\{1\} = v\restriction\{1\} \wedge u(M-1) = v(N-1) = 1\} \\
\quad |[k: int; \\
\qquad S \\
\qquad \{k = (\mathrm{N}t: t\restriction\{0,1,2\} = t: t\restriction\{0,1\} = u \wedge t\restriction\{1,2\} = v)\} \\
\quad]| \\
]|
\end{array}
$$

For $u\restriction\{0,1\} = u$ and $v\restriction\{1,2\} = v$ the *weave* of u and v is the set consisting of all sequences t such that

$$t\restriction\{0,1,2\} = t \wedge t\restriction\{0,1\} = u \wedge t\restriction\{1,2\} = v.$$

We write $W(u, v)$ for the number of sequences in the weave of u and v. The postcondition is then

$$R: \qquad k = W(u, v).$$

Consider sequences s and t such that $s\restriction\{0,1\} = s$ and $t\restriction\{1,2\} = t$. The weave of sequence $s1$ (s extended with element 1) and sequence $t1$ consists of all sequences $w1$, where w is a sequence in the weave of s and t. Consequently,

$$W(s1, t1) = W(s, t). \tag{3}$$

Next, let

$$i = (\mathbf{MAX}h, s': s = s'0^h: h), \qquad j = (\mathbf{MAX}h, t': t = t'2^h: h).$$

($s'0^h$ denotes the concatenation of sequence s' and the sequence that consists of h 0's.) The weave of 0^i and 2^j consists of all sequences of length $i+j$ that have i 0's and j 2's. There are, of course, $\binom{i+j}{i}$ of them. Consequently,

$$W(s, t) = W(s'0^i, t'2^j) = W(s', t') \cdot \binom{i+j}{i}.$$

Furthermore,

$$W(s0, t) = W(s', t') \cdot \binom{i+j+1}{i+1} = \frac{W(s', t') \cdot \binom{i+j}{i} \cdot (i+j+1)}{i+1}.$$

Hence,

$$W(s0, t) = (W(s, t) \cdot (i+j+1))/(i+1). \tag{4}$$

Similarly, we find

$$W(s, t2) = (W(s, t) \cdot (i+j+1))/(j+1). \tag{5}$$

Relations (3), (4), and (5) constitute the crux of our solution. The invariant is

$$\begin{aligned} P: \quad & 0 \leq m \leq M \wedge 0 \leq n \leq N \\ & \wedge\, k = W(u(i: 0 \leq i < m), v(j: 0 \leq j < n)) \\ & \wedge\, i = (\mathbf{MAX} g: 0 \leq g \leq m \wedge (\mathbf{A} h: m-g \leq h < m: u(h) = 0): g) \\ & \wedge\, j = (\mathbf{MAX} g: 0 \leq g \leq n \wedge (\mathbf{A} h: n-g \leq h < n: v(h) = 2): g) \\ & \wedge\, u(i: 0 \leq i < m) \uparrow \{1\} = v(j: 0 \leq j < n) \uparrow \{1\}. \end{aligned}$$

Because the weave of two empty sequences contains one sequence (the empty sequence) P is established by $m, n, k, i, j := 0, 0, 1, 0, 0$.

Postcondition R is implied by $P \wedge m = M \wedge n = N$. Since $u \uparrow \{1\} = v \uparrow \{1\}$ and $u(M-1) = v(N-1) = 1$, we have

$$P \wedge m = M \equiv P \wedge n = N$$

and we can use $m \neq M$ (or $n \neq N$) as the guard of the repetition.

Using (3), (4), and (5), we arrive at the following solution:

```
S:  |[m, n, i, j: int; m, n, k, i, j := 0, 0, 1, 0, 0
    ; do m ≠ M
      → if u(m) = 1 ∧ v(n) = 1 → m, n, i, j := m + 1, n + 1, 0, 0
        □ u(m) = 0 → m, k, i := m + 1, (k * (i+j+1))/(i+1), i+1
        □ v(n) = 2 → n, k, j := n + 1, (k * (i+j+1))/(j+1), j+1
        fi
     od
    ]|
```

The execution time of S is proportional to the lengths of the given sequences.